ÚJFAJTA, AUTOMATIKUS, DÖNTÉSI FA ALAPÚ ADATBÁNYÁSZATI MÓDSZER IDŐSOROK OSZTÁLYOZÁSÁRA

- BŐVÍTETT ÖSSZEFOGLALÓ A 2009-ES VÉGZŐS KONFERENCIÁRA -

Hidasi Balázs
BHIDASI@T-ONLINE.HU

Konzulens:

Gáspár-Papanek Csaba
ügyvivő szakértő, Távközlési és Médiainformatikai Tanszék
GASPAR@TMIT.BME.HU

BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM
VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR
TÁVKÖZLÉSI ÉS MÉDIAINFORMATIKAI TANSZÉK

Budapest, 2009. május 10.

1. Bevezetés

Az idősor-osztályozás problémája napjaink egyik egyre inkább előtérbe kerülő kutatási területe az adatbányászatban. Idősornak tekinthető bármilyen olyan jellegű adat, ami egy/több változó értékeinek sorozata egy/több idő (tér) tengely mentén. Így idősor a tőzsdeindexek alakulása, egy áramkör kimenetén a feszültségértékek sorozata, az EEG felvételek, de akár egy szürke árnyalatos kép is felfogható idősorként, és a példák sorát hosszan lehetne folytatni. Az osztályozás feladata az, hogy ismert osztályváltozójú adatrekordok alapján elkészítsünk egy modellt, amit fel tudunk használni ismeretlen osztályváltozójú adatrekordok megfelelő osztályba sorolására. Az idősor-osztályozás abban tér el a hagyományos osztályozástól, hogy nem rekordokat, hanem idősorokat osztályoz. A jelenleg alkalmazott idősor-osztályozó módszereket két nagyobb kategóriába sorolhatjuk: (1) klasszikus osztályozók, (2) területspecifikus idősor-osztályozók. Az előbbi kategóriába tartoznak a rekordok osztályozására kifejlesztett algoritmusok és ezek átiratai. Erre a csoportra jellemző, hogy az idősort is rekordként szemléli, attribútumként az egyes időpontokban a változó(k) értékeit használva. Ezen megközelítés hátránya, hogy jelentős lehet az adatelőkészítési fázis (pl.: időbeni eltolás kiszűrése, stb.) és nem veszi figyelembe az időbeliséget, ami által sokszor pontatlan modelleket eredményez. A második csoportba tartoznak az egy adott tématerülethez kifejlesztett idősor-osztályozók, amik a saját területükön általában hatékonyak, de más területeken alkalmazva lényegében használhatatlanok.

Az általam kidolgozott algoritmussal szemben négy követelményt állítottam: (1) legyen automatikus, azaz minden egy időtengellyel rendelkező idősort (függetlenül a hossztól, az osztályok számától, a görbék alakjától, stb.) minimális előkészítési munka árán képes legyen osztályozni; (2) legyen pontos, azaz a felépített modell helyesen osztályozzon; (3) legyen magyarázó, azaz a modellt átböngészve legyünk képesek megérteni, hogy mi alapján dönti el, hogy mi legyen az osztályváltozó értéke; (4) legyen általános, azaz az (1)-(3) pontok az alkalmazás bármely területén teljesüljenek.

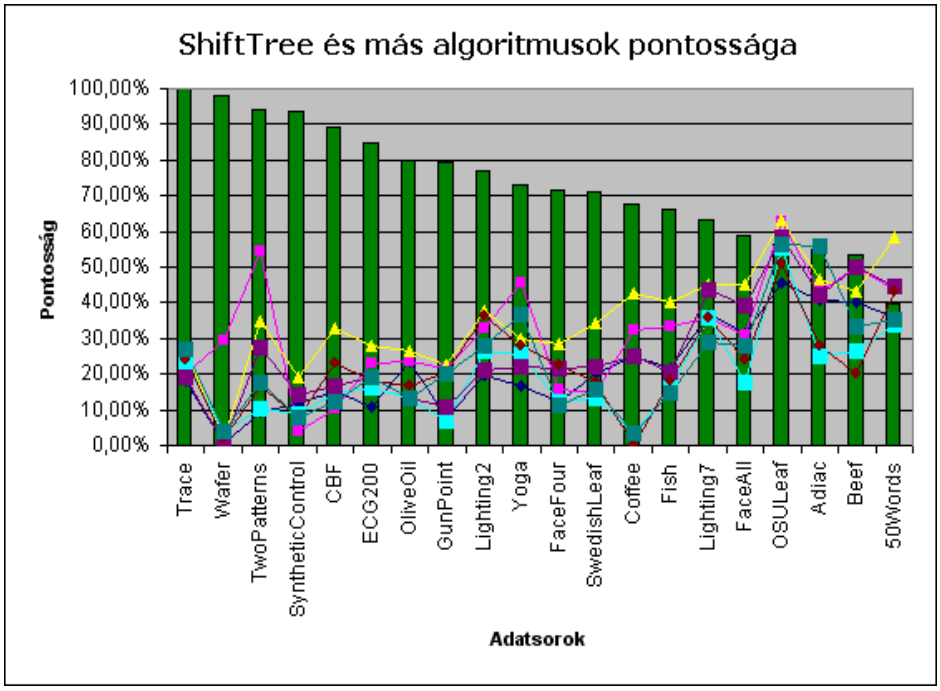
2. A ShiftTree algoritmus

Szakdolgozatomban kidolgoztam a követelményeknek megfelelő *ShiftTree* algoritmust. Az algoritmus a bináris döntési fákat [1] használja alapként, de a csomópontok szerkezetét teljesen átalakítottam. A *ShiftTree* egy csomópontja három funkcionális egységből áll: (1) *Szemtologató* (*EyeShifter – ES*), ami a *szem*nek nevezett pointert mozgatja az időtengely mentén az általa tartalmazott *Szemtologató Operátor*oknak (*EyeShifter Operator – ESO*) megfelelően; (2) *Feltételállító* (*ConditionBuilder – CB*), ami a *szem* helyzete és/vagy az általa mutatott érték és/vagy annak környezete és/vagy a *szem* eltolásának paraméterei alapján egy *származtatott érték*et ad vissza minden egyes idősoron az általa tartalmazott *Feltételállító Operátor*ok (*ConditionBuilder Operator – CBO*) alapján; (3) *Döntő* (*Decider*), ami a benne definiált *jóságérték* függvény alapján eldönti, hogy melyik *ESO*-t és melyik *CBO*-t használjuk a csomópontban, majd kettéosztja az idősorok halmazát a két gyermek csomópont részére. A rendszer lényegében úgy működik, hogy minden idősorra *ESO-CBO* páronként egy attribútumot hoz létre, majd a rekord alapú osztályozókhoz hasonlóan kiválasztja a legrelevánsabbat. A különbség az, hogy itt az operátorok figyelembe veszik az időbeliségben rejlő információkat (pl.: ugrás hossza, mint származtatott attribútum) és emiatt a *szem* az egyes idősorokon akár más időpozícióba kerülhet, ami által az időbeli eltolás jól kezelhetővé válik. A fent leírt elv nagyon általános, a megfelelő operátorok megalkotásával nem csak egy időtengelyes idősorok osztályozására lehetünk képesek, hanem több időtengelyes idősorokat, de akár irányított, címkézett gráfokat is osztályozhatnánk. Ráadásul az alapvető elv miatt az algoritmus tetszőlegesen bővíthető új operátorokkal, amik egy adott területen pontosabb eredményeket adnak, mint az eddig definiáltak, azaz a módszer általános és teljes mértékben testreszabható.

3. A módszer tesztelése

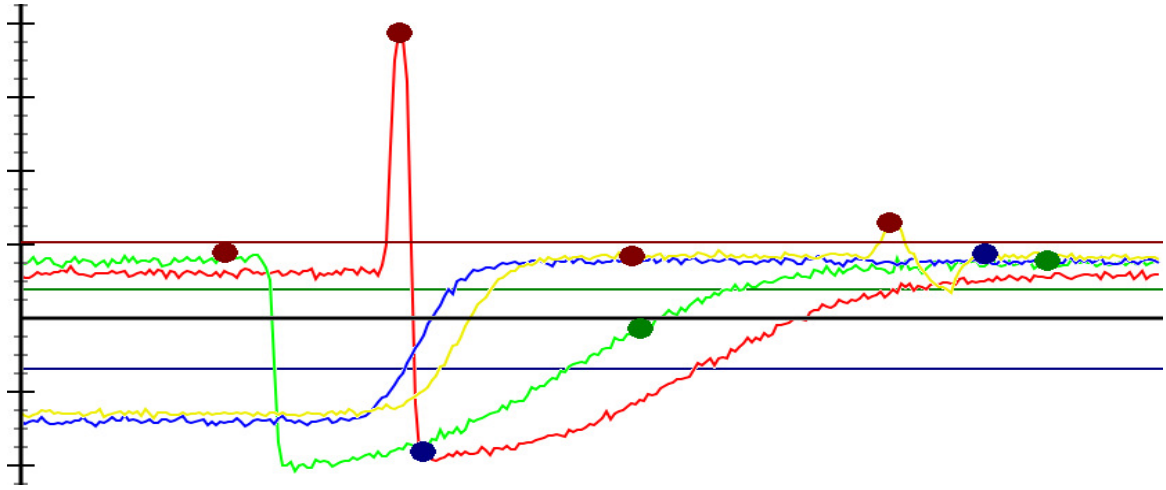
Az algoritmust implementáltam olyan módon, hogy egydimenziós (egy időtengelyes) idősorok osztályozására legyen képes. Egy alapvető operátorkészletet és három *jóságérték* függvényt definiáltam. A szakdolgozatomban részletesen megvizsgáltam az implementáció alapján az algoritmus fontosabb tulajdonságait és hogy mennyiben teljesíti a korábban kitűzött célokat.

Első lépésként az algoritmus pontosságát vizsgáltam meg és hasonlítottam össze más algoritmusokéval (1. ábra). Ehhez a teszthez egy nemzetközi benchmark adatbázist [2] használtam, ami 20 különböző területhez tartalmaz egy-egy idősor-osztályozási problémát. Ezen a 20 problémán más algoritmusok eredményei is elérhetőek. Az összehasonlításhoz minden algoritmusnál az alapértelmezett paramétereket használtam. Az összehasonlítás során azt tapasztaltam, hogy a *ShiftTree* már a legegyszerűbb operátorok használatával is 85%-ban az első helyre kerül a pontossági rangsorban és 90%-ban a top3 algoritmus között van. Viszont néhány esetben, amikor az osztályonkénti tanítópontok száma túl alacsony, akkor hátrébb végez. Ez a tulajdonság a döntési fa alapból öröklődik.



1. ábra: Pontosságok összehasonlítása

A következő teszt a magyarázhatóságot vizsgálta két különböző adatsor segítségével. Az egyik esetben a 2. ábrán látható idősorokat [3,4] kellett osztályozni. Az algoritmus által felépített osztályozó modell szinte olyan volt, mintha egy embert bíznánk meg az elkészítésével. A piros és sárga idősorokat úgy különítette el a kék és zöld idősoroktól, hogy a maximumuk értékét vizsgálta (bordó pontok). A piros és sárga idősorok között ezután úgy tett különbséget, hogy a maximumot követő részen vizsgálta az értékeket (következő lokális maximum, sötétkék pontok). Ez a piros esetén közvetlenül az esés utáni részre esik, a sárga esetén pedig az idősor végére. A kék és zöld idősorok elkülönítése is hasonlóan érthető: a maximumtól egy hosszabbat ugorva vizsgáljuk az értékeket (sötétzöld pontok). Ez a zöld esetén az esés utáni emelkedő részre esik, a kék esetén pedig az idősor végére. A modell a tesztek során 100%-os pontosságot mutatott. A fentiek alapján jól látható, hogy a felépített modellek magyarázóereje magas, maguk a modellek könnyen értelmezhetőek.

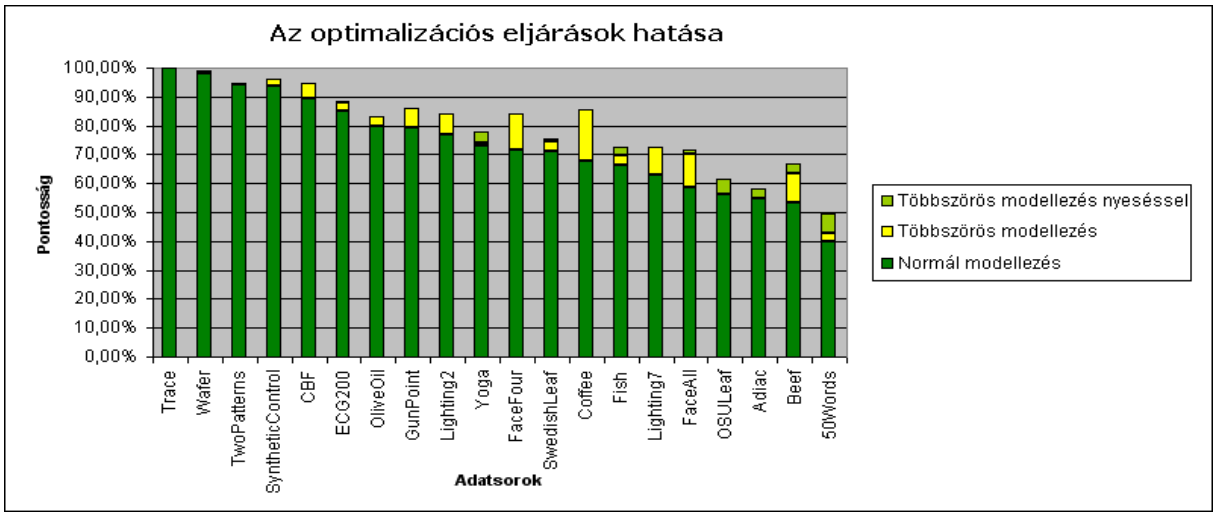


2. ábra: A Trace adatsor egyes osztályainak egy-egy példánya

A tesztelés következő és leghosszabb fázisa annak vizsgálata volt, hogy az algoritmus pontossága és a futási idő hogyan reagál a tanítópontok számának változására. Ezeket a teszteket nem csak a benchmark adatbázis adatain, hanem több többváltozós adatsoron (hangfelismerés, gesztusfelismerés) is elvégeztem, figyelve arra, hogy milyen különbségek jelennek meg az egy- és többváltozós idősorok eredményei között. A tesztek során az összes esetben hasonló jellegű görbét kaptam a futási időre, valamint az egyes adatsorok tanítópontjainak számától függően 4 eltérő csoportba sorolható görbét a pontosságra. Az algoritmus hasonlóan viselkedett a többváltozós idősorok esetén, mint az egyváltozósaknál.

4. Optimalizálás

A modellek értelmezése közben fény derült az algoritmus egy érdekességére: ha a modellezés során több *ESO-CBO* pár is a legjobb jóságértéket adja, akkor az elsőt választja. Viszont sok esetben a tesztelés során pontosabb eredményeket kapnánk, ha egy másik párt választanánk. Ez az észrevétel lehetőséget adott egy optimalizációs eljárás, a többszörös modellezés (*Multiple Modelling – MM*) kifejlesztésére. Az *MM* során nem az első legjobb párt választjuk, hanem az összes ilyet, és egyszerre több modellt építünk párhuzamosan. Ezen modellek közül egy másik idősor-halmaz segítségével tudjuk majd kiválasztani a véglegest. Emellett a túltanulás csökkentése érdekében kidolgoztam egy egyszerű nyesési eljárást, ami mind az eredeti egyszerű modellezéssel (*Simple Modelling – SM*), mind az *MM*-mel képes együttműködni. Tesztekkel bizonyítottam, hogy ez a két (optimalizációs) eljárás tovább növeli az elért pontosságot (3. ábra).



3. ábra: Optimalizálás hatása

5. A dolgozat utóélete

Szakdolgozatomat a 2008/2009 tanév őszi félévében készítettem, mint végzős BSc-s mérnök-informatikus hallgató, és 2009. január 8-án védtem meg a TMIT tanszék záróvizsga bizottsága előtt. Az azóta eltelt időben az algoritmus tovább fejlődött.

Egyrészt alkalmazást előkészítendő, új, bonyolultabb operátorok kerültek megvalósításra. Ezen kívül megvizsgáltam, hogy ha nem az alapértelmezett paramétereket használjuk, hanem előbb megvizsgáljuk az adatainkat és ahhoz állítjuk be a paramétereket, akkor jelentős javulás érhető el a pontosságban. Az alkalmazás során az is kiderült, hogy az összességében kevés – de már a helyes működési tartományba eső – tanítóponttal rendelkező adatsorok osztályozásánál a helyesen megválasztott keresztvalidáción alapuló többségi szavazás jelentősen növeli a pontosságot.

Másrészt lehetőség adódott az algoritmus képességeit egy 2007-es verseny [5] 20 adatsorán tesztelni. A teszt pontosan úgy zajlott le, mintha a versenyre neveztem volna az algoritmussal. Idő hiányában nem volt lehetőségem az adatsorokra külön-külön paraméterezni az operátorokat, így mindenhol ugyanazokat használtam. Még ezzel a megkötéssel is összesítésben az algoritmus a középmezőnyben végzett (5-6. hely a 13 indulóból), ami a kombinált osztályozókat használó mezőnyben nagyon jónak számít. Még jobb az eredmény, ha az adatsoronkénti lebontást nézzük: 6 halmazon messze a legjobb eredményt érte el és 4 esetben nem sokkal maradt el az első helytől (itt a paraméterek finomhangolásával könnyen nyerni lehetne). A többi esetben a tanítóhalmazok kis mérete előre sejtette, hogy nem fog jó eredményeket elérni.

6. Összefoglalás

Összességében elmondható, hogy kidolgoztam egy, az eredeti céloknak teljes mértékben megfelelő, sőt azt néha túlszárnyaló algoritmust. Az algoritmus implementált változatát részletes teszteknek vetettem alá, amikből kiderült, hogy a célok teljesülnek, valamint, hogy pontosság tekintetében az élmezőnybe tartozik. Komplexebb operátorok és megfelelő paraméter hangolás esetén az algoritmus bármilyen területen hatékonyan alkalmazható.

A lehetséges alkalmazási területek köre széles kezdve az orvosi diagnosztizálástól (EKG, EEG hullámok analizálása), az ember-gép interfészeken át (hangfelismerés, gesztusfelismerés, gondolatvezérelt számítógépek), az automatikus tesztelő eljárásokig (alkatrész kimenetének vizsgálata alapján hibás/nem hibás megállapítása).

7. Hivatkozások

[1] **Dr. Bodon Ferenc**: Adatbányászati algoritmusok. (*tanulmány*) URL: http://www.cs.bme.hu/~bodon/magyar/adatbanyaszat/tanulmany/index.html (2008).

[2] **Keogh, E. – Xi, X. – Wei, L. – Ratanamahatana, C. A.**: The UCR Time Series Classification/Clustering Homepage. URL: www.cs.ucr.edu/~eamonn/time_series_data/ (2008).

[3] **Roverso, D**.: Multivariate temporal classification by windowed wavelet decomposition and recurrent neural networks. *In 3rd ANS International Topical Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface*, 2000.

[4] **Chotirat Ann Ratanamahatana – Eamonn Keogh**: Making Time-series Classification More Accurate Using Learned Constraints.

[5] Workshop and Challenge on Time Series Classification. *At SIGKDD 2007.* http://www.cs.ucr.edu/~eamonn/SIGKDD2007TimeSeries.html